UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

| APPLICATION NO. | FILING DATE | FIRST NAMED INVENTOR | ATTORNEY DOCKET NO. | CONFIRMATION NO. |
|---|---|---|---|---|
| 10/688,640 | 10/17/2003 | Antti Kokkinen | 14322US02 | 1958 |

23446 7590 01/18/2008

MCANDREWS HELD & MALLOY, LTD
500 WEST MADISON STREET
SUITE 3400
CHICAGO, IL 60661

| EXAMINER | |
|---|---|
| WANG, BEN C | |

| ART UNIT | PAPER NUMBER |
|---|---|
| 2192 | |

| MAIL DATE | DELIVERY MODE |
|---|---|
| 01/18/2008 | PAPER |

**Please find below and/or attached an Office communication concerning this application or proceeding.**

The time period for reply, if any, is set in the attached communication.

| Office Action Summary | Application No. | Applicant(s) | |
|---|---|---|---|
| | 10/688,640 | KOKKINEN, ANTTI | |
| | Examiner | Art Unit | |
| | Ben C. Wang | 2192 | |

*-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --*

**Period for Reply**

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

**Status**

1)☒ Responsive to communication(s) filed on *14 November 2007*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

**Disposition of Claims**

4)☒ Claim(s) *1-24* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-24* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

**Application Papers**

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

**Priority under 35 U.S.C. § 119**

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

**Attachment(s)**

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

## DETAILED ACTION

1. Applicant's amendment dated November 14, 2007, responding to the Office action mailed August 21, 2007 provided in the rejection of claims 1-24.

Claims 1-24 remain pending in the application and which have been fully considered by the examiner.

Applicant's arguments with respect to claims rejection have been fully considered but are moot in view of the new grounds of rejection – see *O'Neil*, art made of record, as applied hereto.

### *Claim Rejections – 35 USC § 102(a)*

2. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102(a) that form the basis for the rejections under this section made in this office action:

> A person shall be entitled to a patent unless –
>
> (a) the invention was known or used by others in this country, or patented or described in a printed publication in this or a foreign country, before the invention thereof by the applicant for a patent.

3. Claims 1-24 are rejected under 35 U.S.C. 102(a) as being anticipated by Patrick J. O'Neil (WO 02/41147 A1) (hereinafter 'O'Neil' - art made of record)

4. **As to claim 1** (Previously Presented), O'Neil discloses a method for updating software in an electronic device , the method comprising:

- generating an update package for updating at least one software application being generated based upon at least one reference software installed on the electronic device (e.g., Fig. 1A, elements 106 – first version, 108 – second version, and 110 – update package; P. 12, Lines 1-4 - ..., the client device transfer identity information (reference software), including type, model, and/or make of the device, as well as the version of operational software or applications currently in use by the client device; the update generator receives the identity information from the client device and subsequently generates the desired update package required and/or requested by the client device)
- updating the at least one software application using the update package (e.g., e.g., Fig. 1A, element 110 – update package) and the reference software (e.g., P. 12, Lines 1-4 - ..., the client device transfer identity information (the reference software)); and
- wherein the updating leaves the at least one reference software unchanged (e.g., P. 12, Lines 1-4 - ..., the client device transfer identity information (reference software), including type, model, and/or make of the device, as well as the version of operational software or applications currently in use by the client device).

5. **As to claim 2** (Original) (incorporating the rejection in claim 1), O'Neil discloses the method wherein generating an update package for updating the at least one software application based upon the at least one reference software installed on the electronic device comprises:

- accessing a copy of the at least one reference software (e.g., P. 12, Lines 1-4 - ..., the client device transfer identity information (reference software), including type, model, and/or make of the device, as well as the version of operational software or applications currently in use by the client device; the update generator receives the identity information from the client device and subsequently generates the desired update package required and/or requested by the client device);
- retrieving a copy of the at least one software application (e.g., Fig. 1A, elements 106 – first version, 108 – second version); and
- generating an update package (e.g., Fig. 1A, element 110 – update package).

6. **As to claim 3** (Original) (incorporating the rejection in claim 1), O'Neil discloses the method wherein generating an update package for updating at least one software application based upon the at least one reference software installed on the electronic device comprises:

- accessing a copy of the at least one reference software (e.g., P. 12, Lines 1-4 - ..., the client device transfer identity information (reference software), including type, model, and/or make of the device, as well as the version of operational software or applications currently in use by the client device; the update generator receives the identity information from the client device and subsequently generates the desired update package required and/or requested by the client device);

- retrieving a copy of each of multiple versions of the at least one software application (e.g., Fig. 1A, elements 106 – first version, 108 – second version; P. 11, Lines 2-13) and
- generating an update package comprising all transitions between the retrieved versions of the at least one software application (e.g., Fig. 1A, element 110 – update package; P. 11, Lines 2-13 - ... the update generator produces an update package comprising an instruction set which represents a plurality of operations that are desirably used to transform the first original code version into the second updated code version).

7. **As to claim 4** (Original) (incorporating the rejection in claim 1), O'Neil discloses the method further comprising updating multiple update versions of the at least one software application installed on the electronic device is performed using a single update package (e.g., Fig. 1A, element 110 – update package; P. 11, Lines 2-13 - ... the update generator produces an update package comprising an instruction set which represents a plurality of operations that are desirably used to transform the first original code version into the second updated code version).

8. **As to claim 5** (Original) (incorporating the rejection in claim 1), O'Neil discloses the method further comprising installing the at least one software application (e.g., Fig. 1A, element 110 – update package; P. 11, Lines 2-13 - ... the update generator produces an update package comprising an instruction set which represents a plurality

of operations that are desirably used to transform the first original code version into the second updated code version) and the at least one reference software as part of a single installation (e.g., P. 12, Lines 1-4 - ..., the client device transfer identity information (reference software), including type, model, and/or make of the device, as well as the version of operational software or applications currently in use by the client device; the update generator receives the identity information from the client device and subsequently generates the desired update package required and/or requested by the client device).

9. **As to claim 6** (Original) (incorporating the rejection in claim 1), O'Neil discloses the method further comprising updating the at least one reference software (P. 12, Lines 1-4 - ..., the client device transfer identity information (reference software), including type, model, and/or make of the device, as well as the version of operational software or applications currently in use by the client device; the update generator receives the identity information from the client device and subsequently generates the desired update package required and/or requested by the client device) and updating the at least one software application (e.g., Fig. 1A, element 110 – update package; P. 11, Lines 2-13 - ... the update generator produces an update package comprising an instruction set which represents a plurality of operations that are desirably used to transform the first original code version into the second updated code version) as part of a single update.

10. **As to claim 7** (Original) (incorporating the rejection in claim 1), O'Neil discloses the method wherein the at least one software application comprises a plurality of software applications (e.g., P. 5, Lines 3-11 - ... an update generator that compares an image of the first plurality of ... to an image of the second plurality of ... and thereafter generates an update package ...), and the at least one reference software comprises a plurality of reference software (e.g., P. 12, Lines 1-4 - ..., the client device transfer identity information (reference software), including type, model, and/or make of the device, as well as the version of operational software or applications currently in use by the client device; the update generator receives the identity information from the client device and subsequently generates the desired update package required and/or requested by the client device).

11. **As to claim 8** (Original) (incorporating the rejection in claim 7), O'Neil discloses the method further comprising:

- identifying a software application needing updating from the plurality of software applications installed on the electronic device (e.g., P. 18, Sec. of 'Server-Side Update Determination', 1st Par. - .... The update device server analyzes the identity information (reference software at the server side) and checks the server manifest or queries the update store for the presence of the update package. After comparing the available versions of ...., the update store directs the transfer of the update package to the client device);

- identifying whether a reference software corresponding to the software application needing updating is present on the electronic device, wherein if the reference software is not present, then installing the software application and an associated reference software in a single update on the electronic device (e.g., P. 18, Sec. of 'Server-Side Update Determination', 1st Par. - .... The update device server analyzes the identity information (reference software at the server side) and checks the server manifest or queries the update store for the presence of the update package. After comparing the available versions of ...., the update store directs the transfer of the update package to the client device).

12. **As to claim 9** (Original) (incorporating the rejection in claim 7), O'Neil discloses the method further comprising:

- identifying a software application needing updating from the plurality of software applications installed on the electronic device (e.g., P. 18, Sec. of 'Server-Side Update Determination', 1st Par. - .... The update device server analyzes the identity information (reference software at the server side) and checks the server manifest or queries the update store for the presence of the update package. After comparing the available versions of ...., the update store directs the transfer of the update package to the client device); and
- identifying whether a reference software corresponding to the software application needing updating is present on the electronic device, wherein if the reference software is present (e.g., P. 12, Lines 1-4 - ..., the client device transfer identity

information (reference software), including type, model, and/or make of the device, as well as the version of operational software or applications currently in use by the client device), then

- retrieving an update package for the software application needing updating (e.g., P. 22, 2nd Par. - ... using the most current version available (i.e. version 1 updated directly to version 3);
- verifying the update package (e.g., P. 49, 2nd Par. - ... so that the original code is not altered until the update for the code section has been completed and verified thus ...); and
- installing the update package on the electronic device (e.g., P. 6, 1st Par. - ... an update agent ... executes the transformation instructions of the update package thereby transforming the first code version resident in the electronic device into the updated second code version).

13. **As to claim 10** (Original) (incorporating the rejection in claim 7), O'Neil does not explicitly disclose the method further comprising:

- identifying a software application needing updating from the plurality of software applications installed on the electronic device (e.g., P. 18, Sec. of 'Server-Side Update Determination', 1st Par. - .... The update device server analyzes the identity information (reference software at the server side) and checks the server manifest or queries the update store for the presence of the update package. After comparing

the available versions of ...., the update store directs the transfer of the update package to the client device);

- determining if the update is needed immediately; and
- storing the update until the update is needed immediately (e.g., P. 16, 2nd Par. - ... the collector may communicate with both the update server array and ... client devices ... to determine which client devices .... Require updating and if any updates are available. The collector may additionally acquire the necessary updates and , at that time or at later time, distribute them to the client devices ...).

14. **As to claim 11** (Original) (incorporating the rejection in claim 10), O'Neil discloses the method wherein when the update is determined to be needed immediately, then

- invoking an update agent to employ at least the stored update package and reference software (e.g., P. 43, 2nd Par., Lines 12-16 - ... the update agent may also contain the functional logic required to manage the update process ... ); and
- updating the software application with the update package (e.g., P. 6, 1st Par. - ... an update agent ... executes ... of the update package ... transforming the first code version ... into the updated second code version).

15. **As to claim 12** (Currently Amended), O'Neil discloses a system for updating software, the system comprising:

- an electronic device capable of having software installed thereon (e.g., P. 12, Lines 21-24 - … allows updates for software or firmware components in devices ….);
- a software delivery device for receiving and installing a reference software to the electronic device if the electronic device does not have the reference software previously installed (e.g., P. 17, Sec. of 'Client-Side Update Determination', Lines 1-4 - … the client device establishes a communication link with the update store and transfers identify information (the reference software) ….. sued by the client device); and
- the software delivery device receiving and delivering at least one update package (e.g., e.g., Fig. 1A, element 110 – update package) to the electronic device, wherein the reference software facilitates, using the at least one update package, at least one update to application software installed on the electronic device, and wherein the updating leaves the reference software unchanged (e.g., P. 12, Lines 1-4 - ..., the client device transfer identity information (reference software), including type, model, and/or make of the device, as well as the version of operational software or applications currently in use by the client device).

16. **As to claim 13** (Original) (incorporating the rejection in claim 12), O'Neil discloses the system wherein the electronic device further comprises an update agent (e.g., Fig. 8B, element 1025 – Update Agent), the update agent being capable of employing the reference software in conjunction with any retrieved update package to generate updated versions of the application software and also being capable of

updating a plurality of application software employing reference software associated with each application software (e.g., P. 12, Lines 1-4 - ..., the client device transfer identity information (reference software), including type, model, and/or make of the device, as well as the version of operational software or applications currently in use by the client device).

17. **As to claim 14** (Original) (incorporating the rejection in claim 12), O'Neil discloses the system further comprising an update generating system, the update generating system comprising a loader manager, the loader manager:

- managing loading of application software and application software version updates from the software delivery device (e.g., Fig. 8B, elements 1020 – Download Agent, 1025 – Update Agent);
- employing a loader from a loader module (e.g., P. 42, 3rd Par.– the download agent carries out functions related to acquiring the update package while the update agent is responsible for applying … the first original code version is transformed into the second updated code version); and
- employing security services to authenticate software being delivered (e.g., Fig. 12; P. 58, 2nd Par. - … a signature creation and authentication process … is used for security and validation purposes).

18. **As to claim 15** (Original) (incorporating the rejection in claim 14), O'Neil discloses the system wherein the loader manager further comprises an installation agent for installing application software and downloading files from the software delivery device (e.g., P. 42, 3rd Par.– the download agent carries out functions related to acquiring the update package while the update agent is responsible for applying ... the first original code version is transformed into the second updated code version).

19. **As to claim 16** (Original) (incorporating the rejection in claim 16), O'Neil discloses the system wherein the loader manager is adapted to:

- identify an application software needing updating (e.g., P. 12, Lines 14-16 - ... transfer the desired update package to the client device as requested or required);
- identify whether reference software associated with the application software needing updating exists (e.g., P. 18, Sec. of 'Server-Side Update Determination', 1st Par. - .... The update device server analyzes the identity information (reference software at the server side) and checks the server manifest or queries the update store for the presence of the update package. After comparing the available versions of ...., the update store directs the transfer of the update package to the client device); and
- coordinating an update of the application software and an associated reference software in a single update (e.g., P. 22, 2nd Par. - ... using the most current version available (i.e. version 1 updated directly to version 3).

20. **As to claim 17** (Original) (incorporating the rejection in claim 14), O'Neil discloses the system wherein the loader manager is adapted to:

- retrieve the update package (e.g., P. 12, Lines 9-11 - ... may retrieve from .... version of the desired update package);
- access contents of the update package (e.g., P. 8, the controller is configured to sequentially (i) retrieve ..., (ii) apply ..., and then (iii) replace ...); and
- verify the update package (e.g., P. 49, 2nd Par. - ... so that the original code is not altered until the update for the code section has been completed and verified thus ...).

21. **As to claim 18** (Original) (incorporating the rejection in claim 14), O'Neil discloses the system wherein the loader manager is adapted to determine immediacy of a needed update for a particular application software (e.g., P. 16, 2nd Par. - ... the collector may communicate with both the update server array and ... client devices ... to determine which client devices .... Require updating and if any updates are available. The collector may additionally acquire the necessary updates and , at that time or at later time, distribute them to the client devices ...).

22. **As to claim 19** (Original) (incorporating the rejection in claim 12), O'Neil discloses the system wherein the software delivery device is one of a server (e.g., Fig. 1C, element 136a – Update Device Server), a CDROM (e.g., P. 62, Lines 8-15 - ... for

example on a CDROM disk ...), and a network (e.g., P. 11, 1st Par., Lines 13-18 - ... LANs ... WANs ...).

23. **As to claim 20** (Original) (incorporating the rejection in claim 12), O'Neil discloses the system wherein the electronic device is one of a computer, a digital phone, and a digital camera (e.g., P. 11, Lines 18-23 – such as computers, personal digital assistants (PDAs) ... mobile phones ...).

24. **As to claim 21** (Original), O'Neil discloses a method for updating software in an electronic device the method comprising:

- generating a first update package for updating at least one software application, the first update package being generated based upon difference information between first and second software versions (e.g., Fig. 1A, elements 106 – first version, 108 – second version, 110 – update package; P. 5, 1st Par. - ... an update generator that compares an image of the first plurality .... to an image of the second plurality ... and identifies differences between the updated operating code and the resident operating code and thereafter generates an update package ....);
- generating a second update package for updating the at least one software application, the second update package being generated based upon difference information between first and third software versions (e.g., Fig. 1A, elements 106 – first version, 108 – second version, 110 – update package);

- generating a third update package for updating the at least one software application, the third update package being generated based upon difference information between the first and second update packages (e.g., P. 12, 1st Par.,–Lines 6-14 - ... the update generator may be equipped with the ability to generate and provide a plurality of update packages ..., the update generator may create a version manifest, which comprises a list of archived update packages ... and transfer the desired update package to the client device as requested or required); and
- updating the at least one software application using the third update package (e.g., Fig. 1A, element 110 – update package; P. 11, Lines 2-13 - ... the update generator produces an update package comprising an instruction set which represents a plurality of operations that are desirably used to transform the first original code version into the second updated code version).

25. **As to claim 22** (Original), O'Neil discloses a method for updating software in an electronic device, the method comprising:

- generating a first update package for updating at least one software application, the first update package being generated based upon difference information between a first software version and a reference software corresponding to the at least one software application (e.g., Fig. 1A, elements 106 – first version, 108 – second version, and 110 – update package; P. 12, Lines 1-4 - ..., the client device transfer identity information (reference software), including type, model, and/or make of the device, as well as the version of operational software or applications currently in use

by the client device; the update generator receives the identity information from the client device and subsequently generates the desired update package required and/or requested by the client device);

- generating a second update package for updating the at least one software application, the second update package being generated based upon difference information a second software version and the reference software corresponding to the at least one software application (e.g., Fig. 1A, elements 106 – first version, 108 – second version, and 110 – update package; P. 12, Lines 1-4 - ..., the client device transfer identity information (reference software), including type, model, and/or make of the device, as well as the version of operational software or applications currently in use by the client device; the update generator receives the identity information from the client device and subsequently generates the desired update package required and/or requested by the client device);
- generating a third update package for updating the at least one software application, the third update package being generated based upon difference information between the first and second update packages (e.g., P. 12, 1st Par.,–Lines 6-14 - ... the update generator may be equipped with the ability to generate and provide a plurality of update packages ..., the update generator may create a version manifest, which comprises a list of archived update packages ... and transfer the desired update package to the client device as requested or required); and
- updating the at least one software application using the third update package (e.g., Fig. 1A, element 110 – update package; P. 11, Lines 2-13 - ... the update generator

produces an update package comprising an instruction set which represents a plurality of operations that are desirably used to transform the first original code version into the second updated code version).

26. **As to claim 23** (Original), O'Neil discloses a system for updating software, the system comprising:

- an electronic device capable of having software installed thereon (e.g., P. 1, Sec. of 'Field of the Invention' - ... information updating systems .... To a software system and method for updating information .... and distributes the update ....);
- a first update package generator for generating update packages based upon difference information between different versions of software (e.g., Fig. 1A, elements 106 – first version, 108 – second version, 110 – update package; P. 5, 1st Par. - ... an update generator that compares an image of the first plurality .... to an image of the second plurality ... and identifies differences between the updated operating code and the resident operating code and thereafter generates an update package ....);
- a second update package generator for generating update packages based upon difference information between different update packages (e.g., Fig. 1A, elements 106 – first version, 108 – second version, 110 – update package; P. 5, 1st Par. - ... an update generator that compares an image of the first plurality .... to an image of the second plurality ... and identifies differences between the updated operating

code and the resident operating code and thereafter generates an update package ....); and

- a software delivery device for delivering at least one update package generated based upon difference information between different update packages to the electronic device (e.g., Fig. 1A, element 110 – update package; P. 11, Lines 2-13 - ... the update generator produces an update package comprising an instruction set which represents a plurality of operations that are desirably used to transform the first original code version into the second updated code version).

27. **As to claim 24** (Original), O'Neil discloses a system for updating software, the system comprising:

- an electronic device capable of having software installed thereon (e.g., P. 1, Sec. of 'Field of the Invention' - ... information updating systems .... To a software system and method for updating information .... and distributes the update ....);
- a first update package generator for generating update packages based upon difference information between a version of software and a reference software corresponding to at least one software application (e.g., Fig. 1A, elements 106 – first version, 108 – second version, and 110 – update package; P. 12, Lines 1-4 - ..., the client device transfer identity information (reference software), including type, model, and/or make of the device, as well as the version of operational software or applications currently in use by the client device; the update generator receives the

identity information from the client device and subsequently generates the desired update package required and/or requested by the client device);

- a second update package generator for generating update packages based upon difference information between different update packages (e.g., P. 12, 1st Par.,– Lines 6-14 - ... the update generator may be equipped with the ability to generate and provide a plurality of update packages ..., the update generator may create a version manifest, which comprises a list of archived update packages ... and transfer the desired update package to the client device as requested or required); and
- a software delivery device for delivering at least one update package generated based upon difference information between different update packages to the electronic device (e.g., Fig. 1A, element 110 – update package; P. 11, Lines 2-13 - ... the update generator produces an update package comprising an instruction set which represents a plurality of operations that are desirably used to transform the first original code version into the second updated code version).

***Conclusion***

28. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Ben C. Wang whose telephone number is 571-270-1240. The examiner can normally be reached on Monday - Friday, 8:00 a.m. - 5:00 p.m., EST.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on 571-272-3695. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

BCW BW

ERIC B. KISS
PRIMARY EXAMINER

January 15, 2008